RPSLS User Stories

**Out of 65 points**

Using the concepts of OOP by creating classes and using objects (instances of those classes) to interact with each other, create a console version of the classic game Rock Paper Scissors Lizard Spock.

**User stories:**

**(5 points):** As a developer, I want to make good, consistent commits.

**(15 points)**: As a developer, I want to find a way to properly incorporate inheritance into my game.

**(5 points)**: As a developer, I want to account for and handle bad user input, ensuring that any user input is validated and reobtained if necessary.

**(10 points):** As a developer, I want to store all of the gesture options/choices in a List<T>.

**(10 points)**: As a player, I want the correct player to win a given round based on the choices\* made by each player.

**(10 points)**: As a player, I want the game of RPSLS to be at minimum a ‘best of three’ to decide a winner.

**(10 points)**: As a player, I want the option of a single player (human vs AI) or a multiplayer (human vs human) game.

\* Rock crushes Scissors   
Scissors cuts Paper  
Paper covers Rock  
Rock crushes Lizard  
Lizard poisons Spock  
Spock smashes Scissors  
Scissors decapitates Lizard  
Lizard eats Paper  
Paper disproves Spock  
Spock vaporizes Rock

<https://www.youtube.com/watch?v=cSLeBKT7-sM>

Big Picture (main class?): Create RPSLS game so it can be played by 2 players.

classes to big picture: define players, players choice quantity of players, need player choices RPSLS, results of choices (R v P, P v S etc), need game to function, need game to end, results of game.

classes to be added:

define players (string PlayerOne = PlayerOne(name), PlayerTwo = PlayerTwo(name), how to define computer as player? Possibly if choosing single player, this defaults to Player vs computer)

How many players (always min of 2, player 1 vs player 2, or player vs AI/computer) (define variables ??)

need player choices-RPSLS (list) (member variables R, S, P, S, L)

results of choices-(member variables-what beats what when chosen-inherit?)

number of required games (min best of 3, max best of 7. One other choice of best of 5)

start the game ()

results of game/end game(returning winning name. possibly add results if time. )